Colour Tracking for Unintrusive Real-Time Human Motion Capture to Drive an Avatar

Robert Grant
Supervisor: Richard Green

November 15, 2004

Abstract

This report presents two colour tracking techniques developed for use in a real-time interactive motion capture system. The aim of these trackers is to remove the need for special clothing and markers in these systems. The first colour tracker, designed for simplicity and robustness, tracked colours using a static colour model for each of the objects. The second colour tracker, designed to adapt and adjust for colour changes, tracked colours using a dynamic colour model for each of the objects. The dynamic model used the movements of colour in colour space to correct the model to help correctly locate the object.

Acknowledgments

I acknowledge all of the others who worked on the original motion capture project in different areas, including David Sickinger, David Thompson, Billy Chang and Kushal Vaghani. I also acknowledge the HITLab NZ, Story Inc and the Boston Museum of Science for providing me with all the necessary equipment and funding to make this research possible. Finally, I acknowledge Richard Green for his continued support and supervision during the course of this research project.

Published Papers

Grant, R. N. & Green, R. (2004), Tracking Colour Movement through Colour Space for Real Time Human Motion Capture to Drive an Avatar, *in* 'Image and Computing New Zealand, IVCNZ 04, Akaroa, New Zealand'

Contents

List of Figures

Chapter 1

Introduction

Motion capture is becoming main stream in various media, including movies, television and games, and is used primarily for the animation of virtual characters. Historically this has only appeared as prerecorded animation and rarely in interactive systems. This is often because traditional motion capture systems are far too expensive for use outside of big budget productions. These systems often require a large number of expensive cameras surrounding a large stage for the motion capture actor. With improvements in camera quality and low budget motion capture systems being developed with less than five cameras in closer proximities, this obstacle is quickly disappearing. Another reason that motion capture has been slow to enter interactive systems is because of speed. With processor speeds at the level they are now and graphics cards taking most graphics processing away from them, this problem too is quickly dissolving.

One last problem of traditional motion capture remains, this is of markers, body suits and anything a motion capture actor needs to wear to be recognised and have their motion captured by the system. Markers are small objects, often reflective balls placed at certain points of interest on the clothing of a motion capture actor. The cameras recording the actor can identify these balls and use their positions to calculate the 3D skeletal information. Tight monochrome body suits are also often required to be worn with markers to reduce the effects of occlusion and mistracking with an actors clothes. Tracking human bodies without the aid of markers or a suit is a strong area of interest in the computer vision community.

This report presents two colour tracking techniques designed to extend a current motion capture system. This system is designed to capture the movements of a user, and present an avatar to the user that mimics their movements in real-time. The system was designed for public use, so the colour trackers need to rely on unintrusive alternatives to markers. The users are required to hold coloured props and stand in front of a green screen.

The first tracking technique is a static method designed to be robust and simple. It uses a previously configured definition of the tracked colours to locate it in sequential frames. The largest connected group of classified pixels is

considered as the tracked prop.

The second tracking technique is a dynamic method designed to adapt to changes that would not be possible to account for in the previous tracker. Factors such as illumination level, illumination colour and viewing angle were taken into account in this method. It used simple tracking to track and adapt to any movements in colour space of the prop.

1.1 Overview

Firstly, in chapter 2 background research into motion capture and related technologies are discussed. Secondly, in chapter 3 the system is described and discussed, including descriptions of both the static and dynamic colour trackers. Thirdly, in chapter 4 the results of the implemented trackers are presented and then discussed. Finally, in chapter 5 the conclusions and future work are discussed.

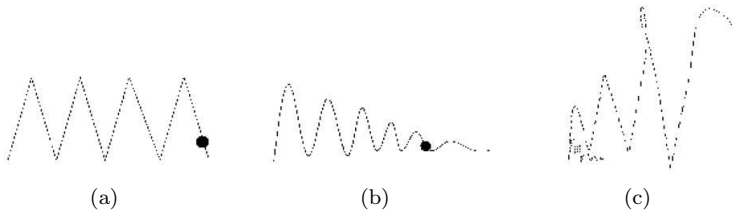
Chapter 2

Background

2.1 Motion Capture

Motion Capture has been around for a long time. In the 1920s, an early form of motion capture, rotoscoping, was invented by the Fleischer brothers to aid animation. Rotoscoping is a technique to aid animation by using live video as a reference and 'tracing' the cartoon characters on top. This gives them a more natural looking motion to an animated character than traditional animation. Now with computers there are techniques to do this automatically with one such method presented by Agarwala et al (Agarwala, Hertzmann, Salesin & Seitz 2004) using edge detection and contour algorithms. Soon it became possible to animate in 3D using the increasing rendering capabilities of computers. This gave animators the ability to create very real looking environments with mobile viewpoints. In terms of character animation, they could now work with an articulated skeleton to create the motions required. To aid this animation, motion capture techniques were developed. Motion capture is the recording of real motion data rather than simulated or created data. While created movement can be made to appear realistic, it is at best an approximation. Simulated

Figure 2.1: An example of a rotoscoped person



(a) (b) (c)

Figure 2.2: (a) shows an example created path, (b) shows a simulated path and (c) is a measured path of a bouncing ball

movement is technically accurate, but will not usually take into account all contributing factors in the real world. Measured movement is more robust and accurate for realism as it does not need the complexity of accurate physics based modeling. These differences can be illustrated with the example of a ball bouncing, as shown in figure 2.2. Geroch (Geroch 2004) describes these differences in more detail. Cameron et al (Cameron, Bustanoby, Cope, Greenberg, Hayes & Ozoux 1997) discus the implications and issues surrounding motion capture and it's use in animation.

The main separation of motion capture technologies are passive and active sensing (Moeslund & Granum 2001). Active sensing uses devices attached to the body which communicate with external devices to derive the location. Devices such as magnetic markers use this kind of sensing and need to be attached by wires. These can be used to track the position and orientation of limbs and joints, or anything they can be attached to (Nickel & Stiefelhagen 2003). Passive sensing uses visible light, infrared or other natural sources reflected off markers to compute the position. While passive is more challenging to compute accurate positions it means the user is not tied to the system in any way. Many computer vision tracking techniques can be employed to make this possible.

2.2 Vision-Based Human Motion Capture

Moeslund and Granum's paper (Moeslund & Granum 2001) contains a comprehensive survey of vision-based human motion capture. It talks about the general structure for systems that analyse human body motion as in figure 2.3. The first component is the initialisation which involves the preparation of the system to receive input. This part of the system mainly concerns camera calibration, adaption to scene characteristics and model initialisation. Nickel and Stiefelhagen (Nickel & Stiefelhagen 2003), in their initialisation stage, use the colour of the highest blob to determine the skin colour for hand tracking. The second component, tracking, usually involves the low-level image and frame to frame processing to prepare data for pose estimation or recognition. This involves such processes as figure-ground segmentation, the separating of a person from the background; representation, how the tracked objects are represented;

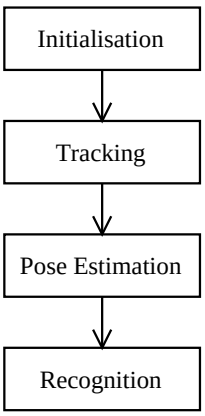


Figure 2.3: A general structure for systems analysing human body motion

and tracking over time, which involves finding corresponding tracked objects in consecutive frames. The third component, pose estimation, is the process identifying how the different parts of a person are configured in the current image. This can be done after tracking using the data received to make estimates about pose, or it can be done as a part of the tracking to utilise other features of the image. The final possible component is the recognition component. Recognition involves the matching of the motion or pose to some kind of action so that the system has some concept of what the user is doing. An example is when a system that tracks the limbs recognises an arm motion by the user as a pointing action, with the system reacting accordingly (Nickel & Stiefelhagen 2003).

Moeslund and Granum also lists reviewed papers in terms of nine abstraction levels; edges, motion, silhouettes, sticks, contours, blobs, texture, depth and joints. These are all features in the tracking component as they deal with image details and locating features within the image, some being higher level than others.

Scott (Scott 2003) describes the motion capture used in the Lord of the Rings movies. This advanced motion capture system used 24 cameras to capture the motion of an actor. The actor needed to wear a skin tight suit with reflective balls arranged on the surface. This method is ideal for movie production and big budget games because it is extremely accurate but too expensive and too intrusive for mainstream use.

The motion capture system presented by Sand et al (Sand, McMillan & Popovic 2003) expands on the marker motion capture technique by using captured silhouettes for more information. They extrude needles from the motion captured skeleton using the silhouette. Then they use this to build a 3D geometry of the captured figure. The result is a motion captured model that includes skin deformations as shown in figure 2.4. This kind of motion capture requires the user to wear a tight black suit, while it is acceptable for recording anima-

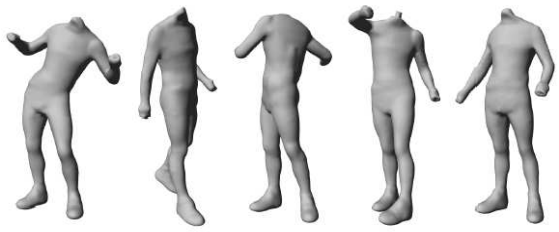


Figure 2.4: Motion capture with skin deformations

tions, the suit is too intrusive for use in an interaction system. Lee et al (Lee, Chai, Reitsma, Hodgins & Pollard 2002) is similar with the restriction of a full body suit.

Guskov et al (Guskov, Klibanov & Bryant 2003) presents a system that tracks arrays of square markers on a glove or t-shirt. The system recognises the squares in arrangement through three cameras consecutively and combines them on another computer to generate 3D surfaces. Through this method the shape of a deforming object such as a hand can be found and recorded. They found that while their system worked well for slow capture, any fast motions reduced the accuracy greatly. They also noted thin limbs were hard to capture as they needed smaller quads to track well, but these are not as reliable to track.

2.2.1 Tracking

Tracking research represents a large proportion of the computer vision literature and is heavily researched. It also has an important place in uninvasive human motion capture as mentioned previously. Approaches to tracking are often split into low level and high level processing. Low level tracking involves using image information such as edges, requiring no high level knowledge of what is in the scene. High level tracking can involve tracking objects such as the head and hands.

Denzler and Niemann (Denzler & Niemann 1997) present a low level approach to tracking. Using the contours of objects, their system could track an object with a 79% success rate. Although their initialisation step only searches for the largest object in the scene at the time, that could potentially be adapted. This algorithm relies on finding a contour that surrounds the object and often only covers the middle section of a pedestrian as seen in figure 2.5.

Heisele et al (Heisele, Kressel & Ritter 1997) presents another form of low level tracking which uses colour to locate objects. It breaks up an image into coloured clusters by repeatedly splitting the image by a statistic on maximum colour differences. This approach results in a segmented image with clusters that can be tracked from frame to frame. Figure 2.6 shows a frame of video taken on a highway, the image has the cluster divisions highlighted to help understand how it works. This method takes advantage of the position of the

Figure 2.5: Low level contour tracking by Denzler and Niemann

colour in the image to improve segmentation.

Nummeriaro et al (Nummiaro, Koller-Meier & Gool 2002) and Vergés-Llahí et al (Vergs-Llah, Aranda & Sanfeliu 2001) both present colour trackers that use various other computer vision algorithms to improve their accuracy. They are examples of using particle filters and histograms to aid tracking. Particle filters involve predicting the likelihood for each position that an object might be in in the next frame. When the tracked object follows a predictable path the algorithm runs quickly, whereas erratic movement can slow it down. Histograms are useful in analysing frames of video as they can reduce information such as intensity or colour from a two dimensional image into a one dimensional graph. They can also be used to translate the image into some other space, such as image space to colour space. Both of the papers used these techniques in novel ways to track and adapt to changes in the colour of an object.

One high level tracking example Pfinder, is presented by Wren et al (Wren, Azarbayejani, Darrell & Pentland 1997). This system segments the image into blobs using statistical models to find the background and foreground areas. These blobs are then found to represent the hands or feet of the user to find some high level representation of the user's limbs as shown in figure 2.7.

Colombo et al (Colombo, Bimbo & Valli 2001) uses assumptions about a person interacting with the system to do high level tracking of their limbs. They assume that the only skin visible on a person will be their hands and head. They also assume that a person's shoes will differ in colour from their pants. This high level form of tracking is mostly limited to one kind of application because it relies on the situation being consistent.

Many other systems that use skin colour tracking also make assumptions about the hands and the head. Such assumptions can be on the colour of skin (Pingali, Tunali & Carlbom 1999) (Gejguš & Šperka 2003) (Wu & Huang 2002) (Yang, Stiefelhagen, Meier & Waibel 1998) or the areas of skin exposed

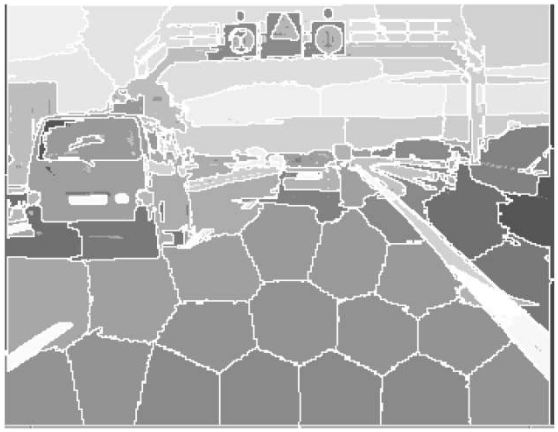


Figure 2.6: Colour/position clustering in a highway scene

Figure 2.7: The Pfinder system segmenting the human body

(Satoshi Yonemoto 2003) (Nickel & Stiefelhagen 2003).

2.3 Motion Capture Interaction

Much recent computer vision research and development is in the area of interaction. With products like the Sony Eye Toy available, vision interaction systems are being seen as possible products. A vision system that could be used for interaction would have to fit a number of requirements. It would have to be:

- fast (operates in real-time)
- robust (track motion sufficiently reliably and accurately so as to enable a compelling interactive experience)
- unintrusive (not require a large amount of preparation before use)
- widely accessible (anyone can use it, regardless of shape, size, colour, etc.)
- cheap (require hardware that is currently mainstream and low cost)
- entertaining or useful (consumers must want to use it)

Nickel and Stiefelhagen (Nickel & Stiefelhagen 2003) presented a system that allowed natural interactivity using pointing gestures. It used computer vision to track the head and hands to extrapolate a line to determine where the user is pointing. While this interactivity seems natural, sometimes having a non-tangible interface can be detrimental to performance. Because the system only reacts when a pointing gesture is recognised the 'pointing tool' is invisible to the user. Colombo et al (Colombo et al. 2001) and Yonemoto et al (Satoshi Yonemoto 2003) presented papers on interactivity using motion capture avatar control. These systems allow the user to control an avatar with their own limbs. While still intangible, there is a more visually continuous and similar link between the user and the avatar that would make it more natural to interact with.

Chapter 3

Technical details

3.1 Aim

The aim of this project is to provide robust colour tracking for uninvasive human motion capture. The colour tracking will be real-time so that interaction with the system can be direct and fluid. The goal is for the tracking to be largely unconstrained by markers and special clothing. Any motion capture system utilising this tracking approach can then be robust to people wearing a wide range of differently coloured clothing. This tracking framework consists of the initialisation and tracking components in figure 2.3. Tracking outputs would include the positions of any colour blobs being tracked and the silhouette of the figure to aid pose estimation.

3.2 Application: The Lord of the Rings Motion Capture Exhibit

The Lord of the Rings motion capture exhibit was a project being developed by the HITLab NZ for the Boston Museum of Science in early 2004. The goals of this motion capture system were:

- to teach users about the motion capture used in the Lord of the Rings
- to provide an entertaining interactive exhibit piece
- to provide a simple motion capture system that did not require markers or special clothing
- to support as many different people as possible, regardless of shape, skin colour, clothing colour, etc
- to create a 'virtual mirror' effect as the users motions are mimicked by a creature of Middle Earth

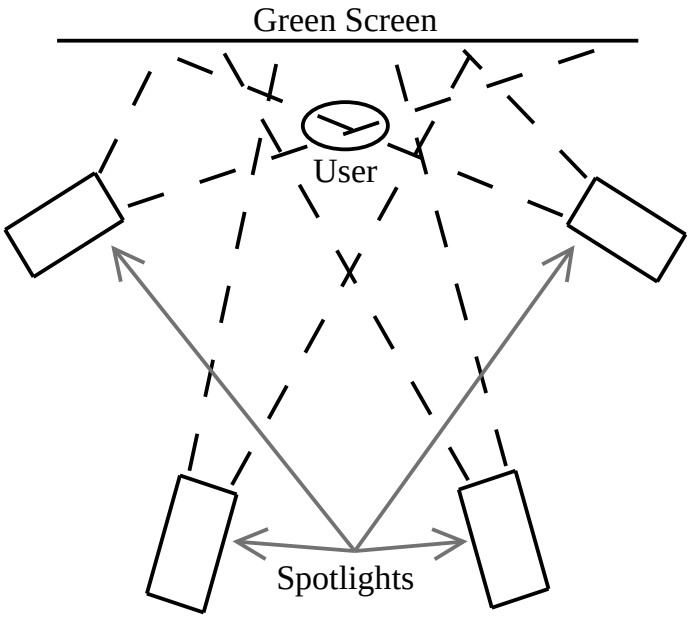


Figure 3.1: Motion capture environment set up

It was known that while the some conditions within the environment would not be ideal, they would be constant. This meant that factors like lighting and background would not change over time. The environment was set up with:

- green screen behind the user to aid motion capture
- two spotlights pointed at the green screen to reduce shadows
- two spotlights pointed at the user to improve colour definition

The set up is shown in figure 3.1 Ideally a bright ambient light source such as a fluorescent or incandescent light would be used, but as the display is located in a science museum there is a restriction on the level of ambient light. Two coloured props, one sword and one shield, were to be held by the user so that the limbs could be unambiguously tracked. The sword had different colours for its' hilt and blade. By introducing props as replacements for markers the system becomes less intrusive to use. However, this also means that with less points being tracked the mimicry will be less accurate.

The initialisation and tracking components of this prior Lord of the Rings system, have been extended by the research presented in this report. However, the Lord of the Rings platform provided this research with a direct application platform that can be used to evaluate its performance, both in speed and robustness. Because of the computational load animating a virtual creature, there is an increased need to implement a computationally efficient colour tracker. This

also means that the tracking implemented can be proven to be real-time and robust enough for use in an unconstrained real world system. It should be noted that although the Lord of the Rings project was jointly developed by this author before the honours project, once the system was completed, during the second quarter of 2004, the work presented in this report used the system as a shell for research to the specific components covered by this report.

3.3 Equipment

The equipment used in this research was as follows:

- Windows PC
  - Intel Pentium 4 2.8GHz
  - 1GB DDR RAM
  - GeForce FX 5900 Graphics Card
- Videre Firewire Stereo Camera
- ADS Turbo USB 2.0 Web Camera (for development purposes)

The software used in this research was as follows:

- Windows XP Operating System
- Microsoft Visual C++.NET 2003
- OpenGL (Graphics Library)
- OpenCV (Computer Vision Library)
- Small Vision System (Stereo Camera Library)

3.4 Design Decisions

It was determined to be advantageous to track colours in the HSV (hue saturation value/intensity) colour space over RGB (red green blue) colour space because of its separation of significant features of a colour. Hue is a particularly useful piece of information about a colour because it 'flattens' the view of the colour into a pure chromatic value. Saturation and intensity are not as stable as hue because they can change with in varying conditions when the light stays constant. Saturation usually decreases when specular reflection is increased on an object, washing out the colour. Intensity usually decreases when diffuse reflection is decreased on an object darkening the colour. It is often helpful to view the HSV colour model as a cylinder as shown in figure **??**. The *rg* (red:green) normalised colour space was also considered because it also separates the chromatic information from the intensity. It does not, however, remove saturation

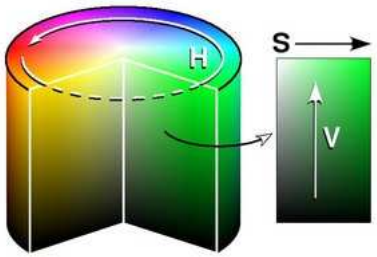


Figure 3.2: The HSV cylinder

information from the colour, and the two dimensional chromatic information makes it more computationally expensive to process compared to the one dimensional hue. To convert an RGB colour into it's HSV representation the following equations can be applied:

$$V = max(R, G, B) \tag{3.1}$$

if $V \neq 0$

$$S = \frac{(V - min(R, G, B)) \times 255}{V} \tag{3.2}$$

otherwise

$$S = 0 \tag{3.3}$$

if $V = R$

$$\frac{(G - B) * 60}{S} \tag{3.4}$$

if $V = G$

$$\frac{180 + (B - R) * 60}{S} \tag{3.5}$$

if $V = B$

$$\frac{240 + (R - G) * 60}{S} \tag{3.6}$$

if $H < 0$

$$H = H + 360 \tag{3.7}$$

A computationally efficient algorithm was used from the OpenCV library which provides a function for converting an image from the RGB to the HSV colour space.

The Small Vision System software package provided an interface with the Videre Firewire Stereo Camera. This software provided the functions to retrieve frames from this camera and then process the frames to produce the 3D information. This meant that pixels would also have a depth value, so this data needed to be included as part of the position of an object. Since the depth

information would often have noise and pixel gaps, the best result would come from an average of all the valid depth values belonging to the object. This could only be done within the tracker part of the system because only it knows the pixels covered by the colour, so it was decided that the tracker would support both 2D and 3D input to maintain flexibility and 3D tracking accuracy.

Various images are also accessible through the tracker component but are not intended for motion capture calculations. These additional images are meant for inclusion in the graphical user interface to provide various representations of the internal functioning of the tracker. These can be used to support education (the first goal of the Lord of the Rings motion capture project from section 3.2), calibration and debugging.

3.5 Static Colour Tracking Procedure

The initial static tracking procedure was incrementally developed. Its primary goals were to be both robust and to run at real-time speeds. To achieve these goals the colour tracker was specified as static. This means that the representations of the colours being tracked would always stay constant. This means that the system only needs be configured once per installation. The amount of time this takes is not important (although should be reasonable). What is important is that minimal configuration is needed per user. To make the tracker useful for a system that is used by many people a day, this needed to be the case.

The process of the tracking will now be covered in the next five sections. These are divided it up into initialisation, image filtering, noise removal, position calculation and configuration.

3.5.1 Initialisation

While the overall system requires a calibration step for each new user to measure arm lengths and the user's height, the colour tracker does not require any user calibration. When the system starts up it loads a configuration file which holds all of the previously configured thresholds required for reliable background subtraction and colour tracking. The process of setting these thresholds is described in section 3.5.5. Once the colours are loaded into the tracker, their tracking can be enabled or disabled. With this ability, the pose estimator (not part of this research) can use the silhouette and colour positions to determine which colours the user is holding. This is done so that a user wearing red, for example, may choose to pick up a weapon with no red and the system will automatically recognise this during the calibration stage. Because this is all done external of the tracker it leaves the system flexible to decide how to determine the correct colours and, in turn, increases the reusability of the tracker. Once the specific colours have been chosen the initialisation is over and the frame by frame processing will begin.

In the first frame, and any frame after, the system can decide to set the background image for the tracker. Once set, the tracker uses the image for the

background subtraction described in the next section. This image should be of the empty area with the green screen behind.

3.5.2 Image Filtering

The first stage of the tracking loop is image filtering. This begins by converting the input RGB image to HSV using the OpenCV function `cvCvtColor()`. All of the converted image's pixels are then iterated through, classifying each as either one of the tracked colours, a background pixel or a foreground pixel. A tracked colour is found when a pixel's HSV colour lies within all of the previously specified ranges for that colour. There is a separate range for the hue, saturation and value/intensity for each tracked colour. Each pixel that classifies as the tracked colour is added to a mask image for that colour.

The background only has a single range for the hue unlike the tracked colours. Background pixels can be classified by either one of two criteria. Firstly, if the background image (retrieved during initialisation) pixel in that location does not fall within the hue range. Secondly, if the current video image pixel in that location does fall within the hue range. In this way the background image acts like a mask for the active area. This means that if the green screen is smaller than the visible area of the cameras, any area outside of the green screen cannot be consided foreground. Any pixel that is background is subtracted from a foreground mask similar to that of a tracked colour. This makes all pixels that are not background part of the foreground, including tracked colours.

3.5.3 Noise Removal

This is done directly after the mask images have been created. Using morphological and connected components functions from OpenCV, a cleaner mask is produced. The foreground mask will likely have a large amount of noise on the background in certain areas. If the green screen does not take up all of the visible area then the edges of it are likely to be subject to more noise than other areas. In darker areas the sensitivity of a camera is less, because of this, shadows are often another cause of noise with colour varying more than usual.

The OpenCV function `cvErode()` removed most of these scattered points by shaving the edges off the mask, shaving small noise out completely and disconnecting any shadows from the figure. The function `cvDilate()` performs the opposite action and called directly after eroding will build the foreground back to it's original size with all the previous noise gone. Finally, the largest group of connected pixels are found in the image using the Teh-Chin chain approximation algorithm. This algorithm traces around the outside of the object. The implementation of this algorithm which was used came from the OpenCV library. These functions find all of the contours in the mask image and using `cvContourBoundingRect()` the bounding rectangles are retrieved. These rectangles are compared and the largest bounding rectangle is considered the largest contour. Although this may not always be correct, it is accurate enough, and fast enough to give it an advantage over the slower pixel counting method. The

resultant contour is then used to overwrite the old mask with a cleaner mask. The foreground mask is now complete and is stored within the tracker for the system to retrieve when appropriate.

This process is repeated on the tracked colour masks with a different combination of erosion and dilation. Because of the number of pixels that were found belonging to the object it was found that an erosion would remove important data. Sometimes parts of a colour object would be slightly divided for various reasons including lighting and shape. It was found that using a dilation alone on the colour often grew the colour enough to connect it, and then finding the largest connected group was enough to remove any noise. Once this new mask was formed it usually covered more of the object than it previously did giving a better result.

3.5.4 Position Calculation

In the final stage of the tracking loop, the tracked colour masks are used to calculate the centres of the objects. This is done by iterating through the mask image and averaging the positions of each set pixel. This is further accelerated by using the bounding rectangles already calculated in the noise removal stage to select a region of interest. When using the stereo camera, a depth value array is iterated through in the same manner as the image. These depth values are tested for being within the accepted range (approximately closer than the background and further than the lens) and then averaged. Once all of the colour positions have been calculated they are stored within the tracker for the system to retrieve when necessary.

3.5.5 Configuration

Configuration is included last because it requires knowledge of the rest of the algorithm to explain. Configuration mode was a separate executable provided in the Lord of the Rings project. It was included to make setting up the thresholds for the colour objects and background easier. The process requires a user to be standing in the view of the cameras with the coloured props. The thresholds are then adjusted to either narrow down on a colour or cut out the background (depending on the task). Another view of the video is provided giving a representation of the tracking without the noise removal stage. This means the configurator can see how much noise could potentially interfere and how much of other areas are falling within the range. The process of configuring is a slow one but given a good configuration can last the entire length of an installation. As mentioned earlier, this is not a big issue for permanent and long term installations with constant environments.

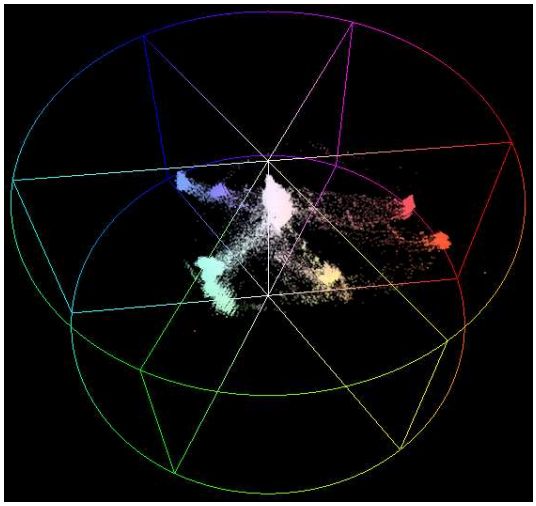


Figure 3.3: Image plotted in an HSV cylindrical histogram

3.6 Dynamic Colour Tracking Procedure

The aim of this research is to improve the tracking algorithm used in the Lord of the Rings system. The specific goal associated with improving this algorithm was to improve the colour tracking by making the colour ranges dynamic. The reason for this is that it can be problematic to find an ideal configuration, make the thresholds too large and other objects will be picked up, make them too small and not enough of the coloured object will be picked up. When a range was not large enough, often the tracker would lose it when it moved into certain areas of the camera's view. Changes in lighting were usually the reason for these slight changes in colour. Even the hue of an object could vary even though in reality it does not change. Including dynamic ranges also allows for configuration to be greatly simplified.

By visualising all of the pixels in an image plotted by HSV colour inside a cylinder, it was easy to see any patterns in colour which emerged. The clusters of points which were formed of the same object were clear to see in the cylindrical plot as shown in figure 3.2. The shape of these clusters was particularly interesting. The most dense part of the cluster was nearer the outside of the cylinder, with a kind of small trail leading toward the centre of the graph, more specifically, toward grey. This means that when white balance is incorrect, these trails from the cluster would point toward where 'true grey' is placed rather than the centre. This can be seen more clearly on a flattened version of the HSV histogram. In the HS histogram in figure 3.3 the distance from the centre of the circle shows saturation, while the angle from the centre shows hue. Lines have been added to show the direction the clusters point, which is in the general direction of 'true grey'. The reason for this is because this 'true grey' is

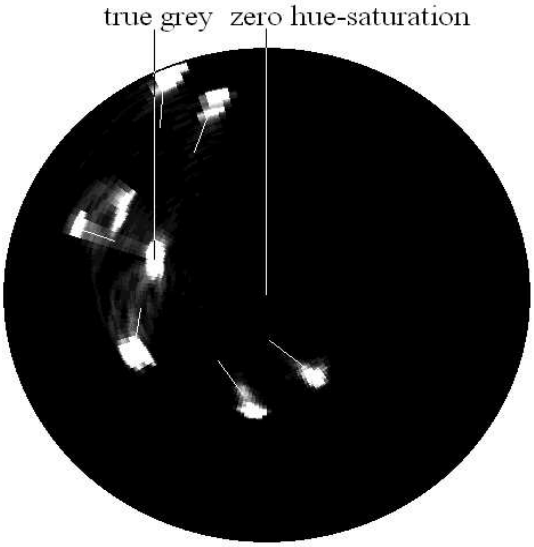


Figure 3.4: HS histogram of a frame that has not been white balanced containing seven distinctive colours and a white/grey region

the illumination colour, and so this means that any changes in the illumination of an object will cause a movement toward or away from this point. This means that if the illumination colour is not positioned at the centre of the graph, then a coloured object can shift in hue, and this is not ideal. Incorrectly positioned illumination also means that hue, saturation and value/intensity ranges will not be able to efficiently cover the long and angled cluster. This is why white balancing is important. Also, sometimes an arc of points between two clusters can be seen. These appear when there are soft edges between objects/colours that touch in the image. The softer the edge the stronger the arc appears, this is a good reason to have the cameras well focused.

The algorithm developed uses only the hue and saturation information from an image. This is to simplify and speed up the process because value/intensity is often the most varying component of HSV colour on an object. Taking advantage of the clustering patterns of object, an algorithm was developed to take advantage of these. The algorithm tries to track these clusters as they vary

The next four sections will describe in detail the steps that the dynamic algorithm takes. This includes initialisation, white balancing, histogram creation, colour adjustment and finally object finding.

3.6.1 Initialisation

Initialisation of the dynamic tracker is much like that of the static one. The only information that is required on each of the tracked colours this time is a centre hue and saturation of the colour. This is the only starting point necessary to

track each of the colours in the image. Because of this, configuration is easier, only requiring the configurator to select directly from a frame of video which objects to track. Also in this stage the R_{shift} and B_{shift} values for white balancing are loaded into the tracker. How the tracker uses these values is described in section 3.6.2

Identically to the static algorithm, colours can be added and removed during runtime. This allows the encompassing system to determine which colours are being tracked and when it should reset the tracked colours back to their default.

3.6.2 White Balancing

Most cameras have some method for automatic white balancing using the following equations.

$$R_{shift} = \frac{G_{max}}{R_{max}} \tag{3.8}$$

$$B_{shift} = \frac{G_{max}}{B_{max}} \tag{3.9}$$

R_{max}, G_{max} and B_{max} are the maximum red, green and blue values out of all pixels in the current image. R_{shift} and B_{shift} are then multiplied with the red and blue values respectively for each pixel in the image. However, this automatic white balancing feature cannot be used in the colour tracker for two reasons. Firstly, to track a colour that can shift constantly would be especially difficult. Initialisation would become much more difficult with the initial colour being uncertain. Secondly, with a green screen and white balancing being gradual, the white balancing algorithm would make the illumination colour green when this is not the colour of the light. Then when a user attempts to interact with the system all other colours will be skewed. Because of this a constant white level is required. This means that any configuration information that is loaded with the tracker will be correct for the white balance. The R_{shift} and B_{shift} values can be calculated during configuration either by the previous equations or by replacing R_{max}, G_{max} and B_{max} with the RGB values of a known white surface in the video (grey can be used, but the brighter the grey, the more accurate the results).

All this step involves is the multiplying of the R_{shift} and B_{shift} values with the red and blue pixels in the image. This results in a better HSV image to work with in later steps.

3.6.3 Histogram Creation

First in this stage, the image is smoothed using gaussian smoothing with the following formula ($\sigma = 1$):

$$f(j,k) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(\frac{-(j^2+k^2)}{2\sigma^2}\right) \tag{3.10}$$

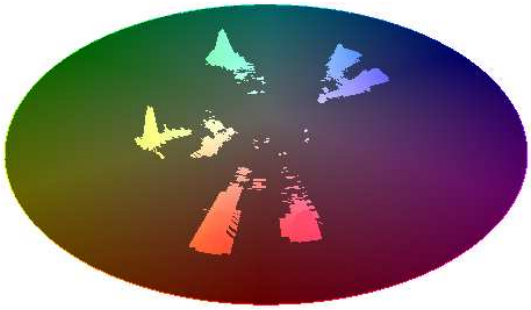


Figure 3.5: 3D view of an HS histogram for an image

This is completed by the OpenCV function `cvSmooth()`. The reason for this is to reduce the effect of noise on the colours in the image. The image then gets converted from the RGB colour space into the HSV colour space using OpenCV functions mentioned previously.

Once this is done, the pixels of the image get accumulated into the bins of a hue-saturation histogram. There are 180 bins for hue and 256 for saturation. These bin sizes can be varied but were kept at their maximum values to maximise the accuracy of the results. This produces a disc histogram as shown in figure 3.4.

3.6.4 Colour Adjustment

This stage is used to approximate the correct ranges for the colour being tracked. It uses the previous frame's colour centre position as a starting position. The algorithm then uses the histogram information to traverse through the bins from this point. By checking the four connected bins around the current bin, it finds the largest bin of the five and changes it to the current bin. Once the steepest ascent algorithm finds the current bin to be the maximum, it ends. This steepest ascent method appears to be robust in finding the most common pixel colour of an object.

After the new cluster centre has been found, the algorithm spreads its ranges out from this point along the h and s axes until a minimum bin size threshold is reached. This can be visually seen as covering the cluster of bins in the histogram that represent the object as in figure 3.5.

3.6.5 Object Finding

The final stage is to locate the object using the ranges previously determined. This step is identical to the steps described in sections 3.5.3 and 3.5.4. It involves creating masks using the HSV values for each pixel by determining if they fall within the ranges for each object. It then uses the erode and dilate morphology functions to remove noise and increase the mass of the objects. The largest

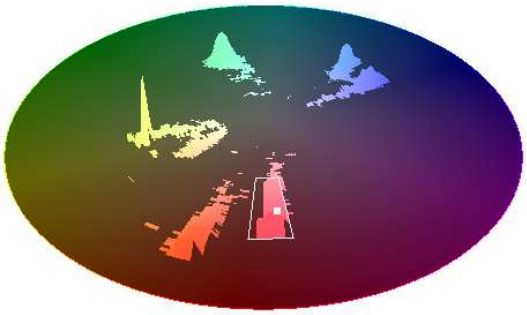


Figure 3.6: Tracked colour region and centre of colour on the HS histogram

connected groups are then found and determined to be the objects. Finally the position of the objects are calculated including the 3D information if relevant.

Chapter 4

Results

4.1 Static Tracker

The static tracker was created with the aim of being as robust as possible while being simple, to keep the system running in real time. To be evaluated the tracker was integrated with the Lord of the Rings motion capture system, this was set up with the specifications listed in section 3.2. The system with the tracker ran at a stable 15 frames per second. This is fast enough to allow real time interactions and a smooth running appearance. After viewing interactions with the system well configured, the tracking appeared to be robust. The tracking was lost on a few occasions when the prop was either moving too fast, angled in certain directions and was moved to the less lit areas of the view. An example of this is shown in figure 4.1. Not much of the source colours were found on the objects, so it found the largest group of connected pixels to be on another area of the image. Despite this, the tracking was robust enough that it provided the other components of the system with reliable input, which helped portray the illusion of mimicry.

4.2 Dynamic Tracker

The dynamic tracker was aimed to be an improvement on the static tracker. Adjusting the colour being tracked as it changes in the image could mean that the where the static tracker previously lost the object (figure 4.1) the dynamic tracker would have adjusted and found a better range for that position. During development, the tracker was tested in a well lit environment with distinctive colours to track, but once integrated into the Lord of the Rings motion capture system the results were not as positive. It was clear that the tracking was unreliable and would often not select the correct area as the object. Even when the tracker did find the object, very little of the object was selected. One effect that was noticeable, and which could have negatively effected the performance of the tracker, was a flickering found in the video. While this flickering effect

(a) (b)

Figure 4.1: In image (a) more than one area is found to belong to the object colour, (b) shows how filtering selects the incorrect areas

was not visible in the video, the histograms could be seen to shift periodically approximately every second. It is believed that this is due to the frequencies of the light sources (connected to New Zealand 50Hz standard mains) and the US configured 60Hz stereo camera system. The speed of the tracker was satisfactory, only reducing the system to about 13 frames per second, which is still fast enough to achieve smooth real-time interaction.

4.3 Comparison

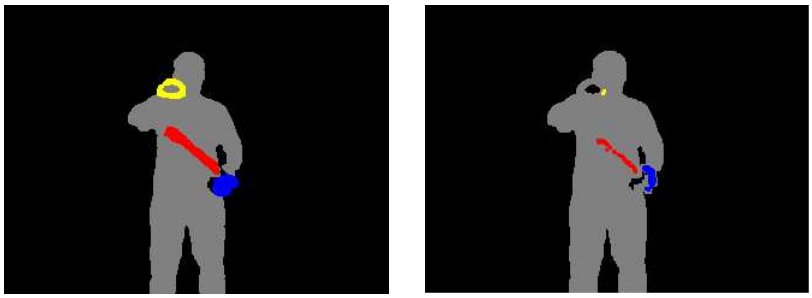
Comparatively both trackers differed substantially. The static tracker performed much better than the dynamic in almost every way. The only area the dynamic tracker succeeded in was it's speed. It may not have been faster than the static algorithm, but the speed drop between the two was small compared to the extra work being done by the dynamic tracker.

Each tracker was tested with two previously recorded videos of two different users interacting with the system. The users in each video were holding two props totaling three colours being tracked, red, yellow and blue. The trackers were both configured for each and then each prop was checked manually frame by frame for correct or incorrect tracking. A successful track was considered one that found any amount of the colour within the prop and none outside the prop. Video 1 contained 229 frames and video 2 contained 306 frames. Table 4.1 shows the results of the test with each of the trackers. From the results shown it is unclear which colour is the best to track. In the case of the static tracker this is a good thing, all colours track consistently successfully. As for the dynamic tracker the results vary largely, between 10.7% and 86.5% with the only variant being the user of the system.

This table only shows part of the difference between the two trackers. The static tracker also finds a much larger proportion of the prop when it is successful compared to the dynamic tracker. Often the dynamic tracker will find only small portions of the prop, this is illustrated by figure 4.2. All of the tracked objects

Table 4.1: Correct tracking rates

Test Video	Prop Colour	Success rate (%) Static	Dynamic
Video 1	Red	98.3	43.7
	Yellow	96.9	10.9
	Blue	98.3	86.5
Video 2	Red	100.0	18.6
	Yellow	100.0	68.2
	Blue	98.4	33.7
Average		98.7	43.6



(a) (b)

Figure 4.2: A comparison between a successful frame from (a) the static algorithm and (b) the dynamic algorithm

appear smaller than they actually are in the dynamic algorithm, while the static provides good coverage, this can especially be seen with the yellow prop.

4.4 Discussion

The dynamic tracker did not perform as well as the static tracker, this does not mean it was a complete failure. Many aspects of the Lord of the Rings motion capture environment hindered its operation. While white balancing should have helped, because of the strong yellow tint of the spotlights and the low sensitivity of the camera in the darkened environment, it was not possible to get clean colour data low in noise. The dynamic tracker relies on the fact that colour from objects cluster together. With large amounts of noise due to the environment coupled with objects being small on the screen, this meant that both the steepest ascent tracking and the low threshold range spreading algorithms would not always produce good results.

Being completely static is what made the static algorithm so ideal in this situation. None of these previously mentioned problems affected it because

the tracker was independent of the environment. As long as the environment stayed in a constant state, the configuration could be made to fit most adverse conditions it was presented with.

Given a better environment, it could be seen that the dynamic tracker could produce better results. This being the case though, the static tracker would likely perform better too. Where the dynamic tracker might perform better, would be in an unevenly but brightly lit environment. To improve the dynamic tracker in adverse conditions, more pre-tracking image processing would need to be done similar to the way white balancing and smoothing helps. The method of range spreading was problematic, if two coloured objects similar in colour lay next to each other on the histogram, then it could spread over to include it. Also if the object was particularly small or had noisy data, then random declines on the histogram could stop it too early. A better way to spread over the colour cluster could prove advantageous to the tracker as a whole.

Chapter 5

Conclusion

It can be concluded that this research achieved it's goal of evaluating two colour tracking algorithms designed to improve the robustness of an existing capture system. With the results of the static tracker, we can see that this was successful in achieving all of the goals that were set. The tracker located the position of the prop successfully at a stable average of 98.7% in the tests, and it found an acceptably large proportion of the prop most of the time. This constitutes the tracker achieving its robustness goal. The other goal, speed, it was also successful in achieving, running at a rate of 15 frames per second when fully integrated with the motion capture system.

The results of the dynamic tracker were not as positive. The speed of the tracker was its largest success, only dropping the systems speed down to 13 frames per second. While it reached a maximum success rate of 86.5%, this varied to as low as 10.9% to show an inconsistant success rate. Also only small proportions of the coloured prop often were identified when tracking. While the dynamic tracker failed in it's aim to improve upon the static tracker by adapting the ranges, it still provided useful information on what factors there are to consider when attempting to do this.

5.1 Future Work

This report has shown two different ways to track colour for use in a real-time motion capture system. Both methods, while largely different in results, could both be developed further to improve their capabilities in speed, robustness or possibly relaxing assumptions. For example, a colour tracker that would work without any assumption of a constant controlled environment would make it ideal for use in a personal environment, such as a form of home entertainment. By gradually pushing past these assumptions we can broaden the scope of its potential uses.

The dynamic algorithm has the potential for much more development. It's failure did not tell us that this approach would not work, only that it needs more

work. Image filters could help highlight some of the information that was noisy and sparse from the input images. Better tracking and spreading algorithms could be developed to better cover the data. Investigation into machine learning algorithms and statistical measures could prove useful in this area.

Bibliography

Agarwala, A., Hertzmann, A., Salesin, D. & Seitz, S. (2004), Keyframe-based tracking for rotoscoping and animation, *in* 'ACM Transactions on Graphics'.

Cameron, G., Bustanoby, A., Cope, K., Greenberg, S., Hayes, C. & Ozoux, O. (1997), Motion capture and cg character animation (panel), *in* 'International Conference on Computer Graphics and Interactive Techniques', pp. 442–445.

Colombo, C., Bimbo, A. D. & Valli, A. (2001), Non intrusive full body tracking for real-time avatar animation, *in* 'International Workshop on Very Low Bitrate Video Coding', pp. 491–500.

Denzler, J. & Niemann, H. (1997), Real–time pedestrian tracking in natural scenes, *in* 'Computer Analysis of Images and Patterns', pp. 42–49.

Gejguš, P. & Šperka, M. (2003), Face tracking in color video sequences, *in* 'Proceedings of the 19th spring conference on Computer graphics', pp. 245–249.

Geroch, M. S. (2004), Motion capture for the rest of us, *in* 'Journal of Computing Sciences in Colleges archive', Vol. 19, pp. 157–164.

Guskov, I., Klibanov, S. & Bryant, B. (2003), Trackable surfaces, *in* 'ACM SIGGRAPH/Eurographics Symposium on Computer Animation', pp. 251–257.

Heisele, B., Kressel, U. & Ritter, W. (1997), Tracking non-rigid moving objects based on color cluster flow, *in* 'Computer Vision and Pattern Recognition, Proceedings', pp. 257–260.

Lee, J., Chai, J., Reitsma, P., Hodgins, J. K. & Pollard, N. S. (2002), Interactive control of avatars animated with human motion data, *in* 'ACM Transactions on Graphics', Vol. 21, pp. 491–500.

Moeslund, T. & Granum, E. (2001), A survey of computer vision-based human motion capture, *in* 'Computer Vision and Image Understanding', Vol. 18, pp. 231–268.

Nickel, K. & Stiefelhagen, R. (2003), Pointing gesture recognition based on 3d-tracking of face, hands and head orientation, *in* 'Proceedings of the 5th international conference on Multimodal interfaces', pp. 140–146.

Nummiaro, K., Koller-Meier, E. & Gool, L. J. V. (2002), Object tracking with an adaptive color-based particle filter, *in* 'DAGM02', p. 353 ff.

Pingali, G., Tunali, G. & Carlbom, I. (1999), Audio-visual tracking for natural interactivity, *in* 'Proceedings of the seventh ACM international conference on Multimedia', pp. 373–382.

Sand, P., McMillan, L. & Popovic, J. (2003), Continuous capture of skin deformation, *in* 'ACM Transactions on Graphics', Vol. 22, pp. 578–586.

Satoshi Yonemoto, Hiroshi Nakano, R.-i. T. (2003), Avatar motion control by user body postures, *in* 'Proceedings of the 11th ACM International Conference on Multimedia', pp. 347–350.

Scott, R. (2003), Sparking life: notes on the performance capture sessions for the lord of the rings: the two towers, *in* 'ACM SIGGRAPH Computer Graphics', Vol. 37, pp. 17–21.

Vergs-Llah, J., Aranda, J. & Sanfeliu, A. (2001), Object tracking system using colour histograms, *in* 'Proceedings of the 9th Spanish Symposium on Pattern Recognition and Image Analysis', pp. 225–230.

Wren, C. R., Azarbayejani, A., Darrell, T. & Pentland, A. (1997), Pfinder: Real-time tracking of the human body, *in* 'IEEE Transactions on Pattern Analysis and Machine Intelligence', Vol. 19, pp. 780–785.

Wu, Y. & Huang, T. S. (2002), Non-stationary color tracking for vision-based human computer interaction, *in* 'IEEE Transactions on Neural Networks', Vol. 13, pp. 948–960.

Yang, J., Stiefelhagen, R., Meier, U. & Waibel, A. (1998), Visual tracking for multimodal human computer interaction, *in* 'Proceedings of the SIGCHI conference on Human factors in computing systems', pp. 140–147.